The Effect Of Java On Network Benefits In Computer Software Markets

Supriya Sarnikar, (E-mail: ssarnikar@wsc.ma.edu), Westfield State College

ABSTRACT

In markets that exhibit network effects it is often believed that society gets inefficiently locked into an inferior technology. This paper utilizes data generated by a natural experiment, afforded by the introduction of the programming language Java, to test whether computer software markets are prone to inefficient lock-in. Findings from a hedonic price regression indicate that Java effectively increased the level of competition in the software applications market and lowered prices of applications software. The regression results are not consistent with the hypothesis of inefficient lock-in.

INTRODUCTION

In a recent antitrust action[1] against Microsoft Corporation, it was argued that the large installed base of Microsoft's Windows operating system virtually guaranteed its continued dominance in the computer operating systems market. The market for computer software is widely believed to be characterized by network effects.[2] Existing literature suggests that network effects lead to de-facto standardization and that the presence of a large installed base often leads to de-facto standardization and excess inertia in the adoption of new and superior technologies. See for eg., Farrell and Saloner(1986)[3], Katz and Shapiro(1986), Arthur(1989) and, Mitchell and Skrzpacz(2006). Anecdotal and experimental evidence is provided in the literature which suggests that the technologies adopted in some markets such as typewriters (David, 1985), Video Cassette Recorders (Cusumano et.al., 1992), nuclear submarines (Cowan, 1990) etc., may have been inefficient because users were locked into inferior technologies in these markets. Liebowitz and Margolis(1990) show that the experimental evidence provided in David(1985) is flawed and present alternative views on welfare impact of network effects in later research (See Liebowitz and Margolis (1995 and 2001)). More recent empirical papers attempt to measure the extent of network effects and explore the subject further (See Gandal, Kende and Rob (2000). But none of these papers test the theory of inefficient lock-in directly or indirectly. However, the theory continues its hold even in recent theoretical papers. See Farrell and Klemperer(2006). Given the policy implications of such beliefs and the ever increasing importance of network industries in the global economy, it becomes more important to subject this theory to empirical testing. Direct tests of this theory, however, are very difficult to conduct as the necessary data are unavailable in most instances.[4] In this paper, data from a recent natural experiment are used and an indirect test of the "inefficient lock-in" theory is constructed.

The focus of this paper is on the network effects in the market for Personal Computer Operating Systems. A computer operating system provides an interface between the computer's Central Processing Unit (CPU) and the

[1] The US Dept. of Justice brought an antitrust action against Microsoft Corporation in 1998.
[2] Network effects are said to exist if the value of a product to a consumer increases with the number of other consumers of the same product. In computer software markets, for example, individual consumers benefit from a bigger 'network' of consumers because they can exchange files with a greater number of other users and also because a greater number of complementary software is available for the bigger network. Other examples of products and services that exhibit network effects are Video Cassette Recorders, CD Players, telephony, e-mail etc.
[3] Farrell and Saloner (1986) show that under some circumstances, "excess momentum" in the adoption of a new technology may also occur.
[4] See Foray(1997) for a detailed discussion of the empirical difficulties in directly testing the theory of inefficient technological lock-in.

various pieces of hardware such as the monitor, printer, etc. Operating systems also control the interaction between various applications software such as word-processing or spreadsheet programs, and the CPU. In general, application software written for a particular operating system cannot run on a different operating system without extensive and costly modifications. Since a huge component of software development costs is fixed, more vendors of software applications prefer to develop applications software for the dominant operating system than the less popular operating systems. Therefore as the network size grows, the number of complementary software compatible with the dominant operating system increases. This effect has been termed as indirect network externalities in Gandal(1994).

Microsoft Corporation currently dominates the market for PC operating systems with its family of Windows operating systems. It is argued that the availability of a greater number of complementary software was crucial for the widespread adoption of Microsoft's operating systems and for its continued dominance in this market (Gandal, Greenstein and Salant 1999). In 1995, Sun Microsystems Inc. introduced a new programming language called Java which posed a potentially serious threat to the dominance of Windows in the operating systems market.[5] Java made it possible, through its Java Virtual Machine (JVM), for programmers to write applications that would run on any operating system. Thus it became possible for developers of application software to write a single program that would run on any operating system enabled with JVM[6] thereby greatly reducing the costs of software development. (In the absence of Java, software developers would have to write separate programs in order to ensure compatibility with each of the different operating platforms.) Netscape Navigator, which was the leading Internet browser at the time, was the major distribution channel for JVM[7]. Even though the need for an operating system could not be eliminated, Netscape and Java together presented an alternative platform to which applications could be written thereby creating greater potential for competition both in the market for operating systems and the market for applications software.

If Java delivered on its promise, consumers using non-Windows operating systems could also have a greater number and variety of application software available to them.[8] Thus Java possessed the potential to eliminate or at least, reduce the indirect network effects in the market for operating systems. However, this promise did not materialize as programs written in Java tended to be much slower in executing than programs written in other languages such as C. In addition, there were security concerns being voiced in the industry regarding Java 'applets'[9]. Despite these initial concerns, Java enjoyed a good deal of support and a steady rate of adoption in all of 1996 and most of 1997[10]. In the second half of 1997 however, Microsoft introduced its own "impure" version of Java that ran better with its own browser and operating system than with the other browsers or operating systems. These actions along with other alleged anti-competitive behavior led to the antitrust trial of Microsoft. Microsoft was accused of using its dominant position in the PC operating systems market to push its own Internet browsing technology. (See Gilbert and Katz(2001), Klein (2001) Whinston(2001), Gilbert (1995), Lopatka and Page (1995) for a history and economic analysis of the various arguments against Microsoft). Microsoft's actions coupled with Sun Microsystems Corp's inability to get Java adopted as an international standard dealt a serious setback to further adoption and development of Java as a cross-platform language. As a consequence, there were only a few software applications written in Java that were commercially available. Most of the applications tended to be custom-written for specific

[5] InfoWorld (Nov.13, 1995) "Java OS; Sun focuses on eclipsing Windows."

[6] Java World (May 1996) " Java jumps to operating systems: All major operating systems to build in support for Java"

[7]InfoWorld May 29, 1995 "Netscape Inks Pact with Sun" – '"Netscape 2.0 came with a built in support for Java. This version became available in late 1995. Until then, Sun's Hot Java browser was the only java-compatible browser."'

[8] See InfoWorld (Oct. 30, 1995) "Smell of Java lures scores of vendors."

[9] "Despite growing support, some security concerns are being raised about Java applets…although Sun maintains that Java includes facilities for screening out viruses, Java applets might open up undetectable security holes on a client computer." InfoWorld Oct.30 1995: 'Smell of Java lures scores of vendors'. "Lord knows, interpreters are slow. Pre-compiling Java source code into portable byte codes saves the time needed for translating syntax, but interpreting byte codes in the Java virtual machine (JVM) is still exceedingly slow compared to the native code produced by a compiler. Thus, Java performance is generally deemed acceptable for small applets but not for any sizable application." Java World, March 1998.

[10] "Software Development '96: Java is this year's hot ticket" Java World, April 1996. "Educators embrace Java: High marks in college signal long-term, real world success of new programming language", Java World, January 1997.

customers. However Sun Microsystems continued to try to improve its technology in order to encourage its adoption.[11]

In summary, it may be argued that even though Java had not yet demonstrated itself to be a superior technology it did have the potential to create more competition in the market for applications software and therefore had the potential to reduce the indirect network effects in the market for computer operating systems. To the extent that indirect network effects are crucial for the continued dominance of an operating system, the weakening of network effects would bring greater competition to the computer operating systems market. Similar arguments have been made by Takahashi and Namiki(2003). This paper however is the first to test whether Java was successful in bringing competition before the alleged anti-competitive actions took place.

MODEL

A random utility model is used to show that the firms whose software products are compatible with the dominant operating system will command a higher price than firms whose software products are not compatible with the dominant operating system. It is argued that the effect of the introduction of Java was (1) to reduce the difference in the extent of network benefits of the Windows vs. non-Windows operating system by increasing the value of the non-Windows network and (2) create more competition both in the application software markets and the Operating System market.

Consider the choice of application software by consumers who have already invested in one of the two operating systems, A or B. Operating system A has a bigger network of users.

Assumptions

1) There are two firms 1 and 2 that produce application software compatible with the dominant operating system A and firm 3 produces software compatible with the operating system B. This assumption is consistent with the observation that more firms are likely to produce software for the larger network than for the smaller network.
2) The software compatible with operating system A is incompatible with the operating system B and therefore cannot be substituted by software that is compatible with the operating system B and vice versa.
3) The greater the variety of complementary products compatible with any product, the higher the network benefit. The benefit from compatibility with the dominant Operating System of network of size S_A is $W(S_A)$ and the benefit from compatibility with the less popular operating system B of network size S_B is $W(S_B)$. By assumption, therefore $S_A > S_B$ and $W(S_A) > W(S_B)$.

The consumers on the operating system A-network have a choice either to buy nothing or to buy application software from firm 1 or from firm 2. The corresponding utilities of each choice are as follows:

$$U_0^A = V^A + \varepsilon_0^A$$
$$U_1^A = W(S_A) + c_1^A - p_1^A + \varepsilon_1^A$$
$$U_2^A = W(S_A) + c_2^A - p_2^A + \varepsilon_2^A$$

Whereas consumers of the operating system B choose between buying nothing and buying application software from firm3. The corresponding utilities of each choice are as follows:

$$U_0^B = V^B + \varepsilon_0^B$$
$$U_3^B = W(S_B) + c_3^B - p_3^B + \varepsilon_3^B$$

[11] "Sun's Hotspot technology promises to deliver interpreted byte codes that run faster than a compiled program" 'Hotspot: A new breed of virtual machine', Java World, March 1998.

c_1^A, c_2^A and c_3^B denote the value of the different product characteristics offered by firms 1, 2 and 3 respectively. V^A and V^B denote the value of waiting another period(s) before buying any software. p_1^A, p_2^A, p_3^B, denote the prices charged by firms 1, 2 and 3 respectively.

The ε's represent consumers' idiosyncratic tastes and are assumed to have a double exponential distribution therefore the difference between the ε's will be distributed according to the logit. The ε^A*'s* are independent of the ε^B*'s*. The probability that an individual consumer chooses firm 1's product is then given by the probability that $U_1^A > U_2^A \text{ and } U_1^A > U_0^A$ which is equal to the probability that the difference between the random component of the utility is greater than the non-random component. The total number of consumers is normalized to one so that the demand for each product is given by the probability that the product is chosen.

Given our assumptions, the demand function for each product is given by:

$$D_1^A = \frac{\exp((W(S_A) + c_1^A - p_1^A)/\mu_A)}{\exp(V^A/\mu_A) + \exp((W(S_A) + c_1^A - p_1^A)/\mu_A) + \exp((W(S_A) + c_2^A - p_2^A)/\mu_A)}$$

$$D_2^A = \frac{\exp((W(S_A) + c_2^A - p_2^A)/\mu_A)}{\exp(V^A/\mu_A) + \exp((W(S_A) + c_1^A - p_1^A)/\mu_A) + \exp((W(S_A) + c_2^A - p_2^A)/\mu_A)}$$

$$D_3^B = \frac{\exp((W(S_B) + c_3^B - p_3^B)/\mu_B)}{\exp(V^B/\mu_B) + \exp((W(S_B) + c_3^B - p_3^B)/\mu_B)}$$

μ_A and μ_B are parameters of the logit distribution and are positive constants.

Given the above consumer demand functions, the firms choose prices simultaneously to maximize their individual profit functions, which are denoted by π below. 'F' denotes the fixed costs of developing software.

$$\pi_1^A = p_1^A D_1^A - F_1^A; \quad \pi_2^A = p_2^A D_2^A - F_2^A; \quad \pi_3^B = p_3^B D_3^B - F_3^B$$

The first order conditions yield the following implicit relationships for the prices.

$$p_1^A = \mu_A\left[1 + \frac{\exp((W(S_A) + c_1^A - p_1^A)/\mu_A)}{\exp(V^A/\mu_A) + \exp((W(S_A) + c_2^A - p_2^A)/\mu_A)}\right]$$

$$p_2^A = \mu_A\left[1 + \frac{\exp((W(S_A) + c_2^A - p_2^A)/\mu_A)}{\exp(V^A/\mu_A) + \exp((W(S_A) + c_1^A - p_1^A)/\mu_A)}\right]$$

$$p_3^B = \mu_B\left[1 + \frac{\exp((W(S_B) + c_3^B - p_3^B)/\mu_B)}{\exp(V^B/\mu_B)}\right]$$

Predictions from the model are as below.

(1) $\frac{\partial p_1^A}{\partial c_1^A} > 0$, $\frac{\partial p_2^A}{\partial c_2^A} > 0$, $\frac{\partial p_3^B}{\partial c_3^B} > 0$; Therefore, as quality improves the price of the software increases, *ceteris paribus*.

(2) $\dfrac{\partial p_1^A}{\partial W(S_A)} > 0,\ \dfrac{\partial p_2^A}{\partial W(S_A)} > 0,\ \dfrac{\partial p_3^B}{\partial W(S_B)} > 0$; Therefore, as the size of the operating system network increases, *ceteris paribus*, the price of the application software increases.

(3) $\dfrac{\partial p_1^A}{\partial V^A} < 0,\ \dfrac{\partial p_2^A}{\partial V^A} < 0,\ \dfrac{\partial p_3^B}{\partial V^B} < 0,$

As the value of waiting increases, *ceteris paribus*, the prices of all applications software on either network will decrease. It is argued that the introduction of Java and the possibility of cross-platform applications becoming available in the future should have increased the consumers' value of waiting in both networks. Therefore the potential for increased competition in the future should have depressed prices of all platform-specific applications software regardless of the operating system network[12]. In order to induce the customers who value cross-platform applications to buy platform-dependent software now, firms would have to lower prices considerably. Software firms would have a strategic incentive to lower prices on existing prices of platform-specific software because if they manage to capture a larger share of the existing applications market they would then have a better chance of capturing the newly emerging cross-platform market as well. This is because users experience significant switching costs in learning to use new software. For example, if one is used to the interface and functionality offered by WordPerfect then, as an end user, one would prefer to use WordPerfect whether working on a machine that is running Microsoft Windows, IBM's OS/2 or any other operating system. That is, most end users do not care about the operating system they are running but only about the applications that are available for the operating system. Thus, even though a customer may value cross-platform capabilities of an application, it would still prefer applications that look, feel, and function just like the applications that the customers used before cross-platform capability became a possibility.

EMPIRICAL ANALYSIS - HEDONIC PRICE REGRESSIONS

To test whether the introduction of Java actually resulted in lower prices of platform-dependent software applications, a quality-adjusted hedonic price regression model similar to Gandal (1994) is used. The data correspond to the time period 1993-1997 and focus on the market for graphics software applications. Graphics software applications were chosen for several reasons. First, graphics applications were sold largely as stand-alone products during the time period under consideration. This is important because it helps to separate the effects of bundling from the effects of pure price competition. A second reason for choosing graphics software was that all the graphics applications offered compatibility with all the standard graphics file formats. This meant that there were no significant direct network externalities within the graphics market itself and there was no clear leader or dominant product among the various graphics software programs during this time frame. A third reason is that Microsoft was not a major player in this market. Entry in markets where Microsoft was vertically integrated such as the word processing market was more difficult and less likely but in markets where Microsoft lacked a strong presence such as the graphics applications market, entry was more likely (See Liebowitz and Margolis 2001). Microsoft gained a presence through the acquisition of Corel graphics in 2003 but during the time period in this study, Microsoft lacked a significant presence.

The empirical model is a regression of the log of the real price of the graphics applications on a number of independent variables that measured the quality and performance characteristics of the applications. Time dummies control for any time specific effects and dummies representing each firm in the dataset control for any firm fixed effects. The installed base of the operating system that the software application program is primarily written for, served as a measure of the size of the operating system network. The coefficient on the installed base variable, if significant, would be an indicator of whether the application providers were able to capture any premium for compatibility with the dominant operating system. The introduction of Java increased the *potential* for

[12]Even though the value of waiting another period may be higher for consumers on network B, it is not clear whether this would result in the prices of B-compatible software falling more than A-compatible software.

competition in both the operating systems market and the software applications market. In the applications market it would have increased the value of the option to wait for cross-platform applications. Therefore, as shown in the previous section, it would result in a fall in the prices of platform-specific applications on both operating systems.

DATA

Data on the price and quality characteristics of graphics application software was compiled from product reviews published in PC magazine, InfoWorld, PC World, PC Computing, Compute!, Windows magazine and OS/2 e-zine for the years 1993-1997. Graphics software comes in different packaged forms – some are written for very specific purposes such as illustration, image viewing and editing, desktop publishing (DTP), charting etc. while others include two or more of these functions. The data correspond to all of these programs and every effort was made to control for all the features that described these software packages. The resulting sample is an unbalanced panel with a total of 107 observations out of which 13 were compatible with OS/2, and the rest with the Windows operating system. There were 42 distinct software products. Whenever multiple versions (such as standard and professional) were available for any product in the same year, each version was treated as a separate product i.e. each version is a separate observation within the same year. Version upgrades (such as version 1.0 and version 2.0) are treated as the same product i.e., whenever two versions (old and new) were available in the same year, only the new version appears in the sample in that year. If no new version was available in the subsequent year(s) but the old version continued to be sold and was reviewed in any of the above mentioned magazines, then the old version appeared in the sample in the subsequent year(s) also as it was a choice available to consumers in the subsequent years. Such observations however were few and the results presented are not sensitive to the inclusion of the same. The following is a list of all the variables in the empirical model.

Dependent Variable

RPRICE: is the list price of the product in real dollars. As the street price was not available for all the products in the sample, the list price was used even though it does not represent the ultimate cost to consumers. However, this should not be of great concern since the list price is found to be very highly (over 99 percent) correlated with the street price and typically the discount, which was typically found to be about 30-55 percent, was found to be uncorrelated with the characteristics of the software. The natural log of the real list price is the dependent variable in this model.

The independent variables include a list of quality characteristics, a measure of network size, a variable that is constructed to measure the impact of Java and a list of variables that control for fixed time and firm effects.

Independent Variables

Measures Of Quality

- **CLIPART**: This dummy variable takes on a value of 1 if the package comes with a library of pre-selected pictures that may be useful in constructing illustrations. This feature is most useful for users who are not professional graphics designers.
- **MULTIMEDIA**: The variable MULTIMEDIA takes on a value of 1 if the package offers support for sound, video and/or animation, zero otherwise.
- **TEXT**: The variable TEXT takes on a value of 1 if the software is capable of importing either formatted or unformatted (ASCII) text. All software had some capability of inserting text onto an image but not all could import text. This feature is most useful for desktop publishing tasks.
- **DATA**: This variable takes on a value of 1 if the program can import data from at least one of the popular data storage and manipulation programs such as Excel, Lotus, and DBase.
- **RAM**: This is a continuous variable representing the minimum amount of Random Access Memory (RAM) that was required to run the program. This was included to control for the hardware requirement standards that might influence consumer's demand for any software. Also generally speaking, it is observed especially for graphics applications that the more features a program has, the more RAM it requires to run

smoothly. Therefore, this may also serve as a proxy for the many quality characteristics that cannot be easily measured.

- **CHARTS**: This variable takes on a value of 1 if the program is capable of creating at least basic charts such as a bar, pie and line graphs etc.
- **OLE**: This variable takes on a value of 1 if the program supports Object Linking and Embedding (OLE). This feature enables active links between different programs- thus if the data underlying a chart is changed in the spreadsheet program, it is automatically reflected in the linked chart. This feature is only available for Windows compatible software. However since not all Windows compatible software included this feature, it is important and possible to control for the feature.
- **DDE**: This variable takes on a value of 1 if the program supports Dynamic Data Exchange (DDE). This also is a feature that is available only for Windows compatible software however all not Windows compatible software provide it.
- **DRAW:** is a dummy variable that takes on the value of 1 if the software is primarily meant for illustration purposes and therefore provides specialized and sophisticated drawing tools that are generally unavailable in other graphics applications.
- **EDIT:** is a dummy variable that takes on a value of 1 if the software is primarily meant for viewing and editing photos/images.
- **DTP**: (Desktop Publishing) is a dummy variable that takes on a value of 1 if the software consists of specialized tools for publishing activities.
- **PRES:** is a dummy variable that takes on the value of 1 if the software provides special presentation tools.

Measures Of Compatibility

- **File format standards**: The dummy variables **BMP, EPS, GIF, JPEG** and **TIFF** take on the value of 1 if the package offers full/partial compatibility with the respective formats. Compatibility with a particular format allows the user to exchange files with other software applications that support the format. Graphics image formats fall in two broad categories – bitmap and vector. Bitmap images store information in the form of dots and pixels whereas vector images are collections of vectors and shapes. Of all the file formats listed above, .EPS is a vector format whereas the others are bitmaps. The .GIF and .JPG (JPEG) formats are most used for web publishing. An alternative variable (**BITMAP**) was also constructed, which took on the value of one if the application provided compatibility with all of the four bitmap formats listed above and zero otherwise.
- **WIN95**: This variable indicates compatibility with Windows95 operating platform. Microsoft released Windows95 in August/September of 1995 even though it had pre-announced this new improved upgraded version much earlier. Windows95 was a 32-bit operating system that would have made incorporation of many additional quality features possible for software applications. This variable was therefore included to account for the improved quality features that it may have made possible. The coefficient on this variable therefore is expected to be positive.
- **INST_BASE**: is a continuous variable that is a measure of the installed base (in millions of units) of the operating system in any given year. For example, if the observation corresponds to a program that is compatible with the Windows operating system in the year 1993, then the variable takes on the value of the installed base of Microsoft's operating systems in 1993. The sample considered consists of software written for the OS/2 and both Windows3.1 and Windows95 platforms. The installed base data for 1992 was obtained from extracts of International Data Corporation's (IDC) reports published in various software magazines. The installed base data for the rest of the years was constructed from the market share and shipments data in the now publicly available trial exhibits from the Microsoft Antitrust case.

Fixed Effects

- **TIME**: The time dummies **YR_93, YR_94, YR_95, YR_96, YR_97** take on a value of 1 if the data correspond to that year. These dummies serve as a control for any market-wide or economy-wide time-specific exogenous factors that might have impacted the prices of the software under consideration.
- **FIRM:** The data correspond to graphics application software that was developed by either of 22 firms. The firm dummies serve as a control for any number of unobserved characteristics that are unique to a firm and/or its product (e.g. the product interface, ease of use, organizational culture etc.). In addition, the dummies also serve as controls for the firm's market share and resultant network effects within the graphics applications market. As there was no single firm or product that was a clear leader in all the segments of this market during the time period under consideration, it is reasonable to assume that this is an adequate control for the direct network effects, if any, within the graphics software market.

The Effect Of Java

- **JAVA:** is a dummy variable that takes on a value of 1 only if the software was introduced after October 1995. It is important to note that this variable is *not* simply equal to the sum of the 95, 96 and 97-time dummies. This was because (1) this variable takes on a value of one for only a subset of the observations in year 1995 and (2) because I have observations for versions of software applications that continued to be sold after October 1995 but were actually introduced before October 1995 and so these observations had a value of zero for the JAVA dummy. Typically firms introduced new versions of their product on a cycle of one to two years. However there were some applications that did not see new updated versions for the entire period under consideration. The sample includes observations on all software that was available for sale and that was reviewed during the time period under consideration even if the versions were old. To check for robustness of results the regressions were estimated without the old software as well. The results were not sensitive to including the old software.

RESULTS

The descriptive statistics are shown in the appendix Table 1. Table 2 shows the regression results. The first column of Table 2 shows the parameter estimates for the full model. The quality and performance characteristics captured by the measures of text and data handling capabilities are positive and statistically significant. On average, the software that provided text capability was priced 42 percent higher than software that did not provide such capability. Similarly data handling capabilities increased prices by almost 30 percent. The results also show that the memory requirement RAM serves as a proxy for performance features that are hard to measure and on average every additional megabyte of RAM required resulted in a price increase of about 1.1 - 1.4 percent. The negative coefficient on multimedia is baffling but this variable was highly correlated with RAM and therefore the coefficient on RAM should be appropriately interpreted in conjunction with the coefficient on Multimedia capability.

The coefficient on installed base is statistically insignificant suggesting that the applications providers are unable to charge any premium for compatibility with the Windows platform. This result even though unexpected, is not unreasonable as it is possible that the application providers are unable to charge a higher premium for the bigger network since they also face greater competition in the bigger market. This suggests that any compatibility premium probably goes entirely to the producer of the dominant operating system with the application providers unable to capture any significant premiums due to existing and potential competition.

The coefficient on the JAVA term is negative and statistically significant at the one percent significance level. This suggests that even though software providers were already unable to charge any premium for compatibility with the dominant operating system, the introduction of Java further eroded their price-markups by increasing the possibility of more competition in the future. The increase in competition is also evident in the number of software written for OS/2 after 1995. Even though these applications were not cross-platform capable, there was a rise in the number of software written exclusively for OS/2 as is evident from our data (See Table 1 for descriptive statistics).

The least squares dummy variable approach taken here to estimate firm fixed effects leads to a high level of multi-collinearity. The Belsley-Kuh-Welch condition index was found to be between 40-80. Such collinearity among the regressors is not unusual in hedonic regressions where most of the regressors are dummy variables. The problem is aggravated due to the additional dummies that control for firm fixed effects. Due to the three-way error components (firm, product and time components in the residual error term) in this model however, it was not possible to use the usual fixed effects estimator. As the data collected corresponds to almost the entire population of graphics applications, it is not possible to further enhance it. Another remedy is to drop the variables that are highly collinear and/or most insignificant from the model. The results show that the file format standards in general are highly collinear and not an important determinant of the product's price. But instead of dropping them from the model, they were combined into a single measure of compatibility. In column 3 of Table 2 the variables BMP, GIF, JPEG and TIFF were dropped from the model and instead included the variable BITMAP (The construction of this variable is explained above in the variable descriptions). The results show that compatibility with all the four file format standards increases the price by almost 26 percent. The other parameter estimates do not change very much but the standard errors are lower thereby improving efficiency of the estimates. The condition index falls but still remains beyond the recommended level. I therefore re-estimate the model by dropping the variables that are most insignificant (EPS and Clipart). Column 5 of Table 2 shows the results when the least significant variables are dropped from the model. The precision of the estimates improves slightly and the condition index drops further (from 57 to 44) and is closer to the acceptable level of 30.

The column 7 of Table 2 shows regression estimates with the variable WIN95 included. The inclusion of this variable increases the overall condition index and lowers the efficiency of estimates. The variable itself is statistically insignificant and therefore we may conclude that the negative coefficient on JAVA is indeed a result of greater anticipated competition rather than a coincidental effect of the almost simultaneous introduction of Windows95. I expected the effect on prices of Windows95 to be positive due to the improved quality features. Most of the firms in the sample introduced newer versions for Windows95 simultaneously with the introduction of Windows95. Therefore, any drop in prices due to consumers waiting for a Windows95 version would have occurred prior to the introduction of software compatible with Windows95 i.e. before August 1995. Therefore the drop in prices observed after the introduction of Windows95-compatible software could be reasonably attributed to the introduction of Java.

The consistent negative effect on prices from the Java variable therefore, indicates that Java was indeed effective in increasing the potential for competition in the market for graphics software applications and therefore in reducing the indirect network effects in the PC-operating system market.

CONCLUSION

This paper contributes to the scarce empirical literature on network effects and provides an indirect test of their impact on welfare. The results from the hedonic price regressions suggest that the potential availability of a platform-independent software application helped to decrease the prices of all graphics software. The results therefore suggest that despite the demand-side and supply-side economies of scale that are thought to exist in the software markets in general, these markets are quite competitive. The significant increase in the number of software applications available for the OS/2 platform after the introduction of Java and the reduction in prices of all graphics software suggest that Java was indeed successful in reducing the indirect network effects in the PC operating systems market. Given the hypothesized importance of indirect network benefits in the PC operating system market, the results might be interpreted to suggest that the PC operating system market might also be subject to competition. Future research would directly test whether or not positive feedback is indeed present in the PC operating systems market and whether such feedback leads to inefficient market outcomes.

ACKNOWLEDGEMENTS

Supriya Sarnikar, assistant Professor, Department of Economics and Management, Westfield State College, MA. An earlier version of this paper was presented at the "Helsinki Conference on Antitrust Issues in Network Industries". I am grateful to Stanley Reynolds, Price Fishback, Ronald Oaxaca, Neil Gandal, Daniel Houser and

John Wooders for valuable suggestions and comments. Tobias Kretschmer, seminar participants at the University of Arizona and conference participants at Helsinki provided constructive comments on earlier drafts. Thanks to Surendra Sarnikar and Ram Pudupet for helping me gain a basic conceptual understanding of computer operating systems and Java. Any remaining errors are of course mine.

REFERENCES

1. Arthur, W. Brian (1989) Competing Technology, Increasing Returns, and Lock-in by Historical Events *Economic Journal*, Vol 99, 116-139
2. Cowan, R. (1990) Nuclear Power reactors: a study in technological lock-in. *The Journal of Economic History*, Vol. 5(3), 541-567
3. Cusumano, M., Y. Mylonadis and R. Rosenbloom (1992) Strategic Maneuvering and mass-market dynamics: the triumph of VHS over Beta *Business History Review*, Vol 66, 51-94
4. David, P.A. (1985) Clio and the Economics of QWERTY *American Economic Review*, Vol.75, 332-337
5. Farrell, J. and G. Saloner (1986) Installed Base and Compatibility: Innovation, Product Preannouncements, and Predation *American Economic Review*, Vol. 76, 940-955
6. Farrell, Joseph; Klemperer, Paul (2006) Coordination and Lock-In: Competition with Switching Costs and Network Effects C.E.P.R. Discussion Papers: 5798-2006
7. Foray, Dominique (1997) The Dynamic implications of increasing returns: Technological change and path dependent inefficiency *International Journal of Industrial Organization*, Vol.15, 733-752
8. Gandal, N.M., M. Kende, and R. Rob (2000) The Dynamics of Technological Adoption in Hardware/Software Systems: The case of Compact Disc Players. *RAND Journal of Economics,* Vol. 31(1), 43-61
9. Gandal, N.M., S. Greenstein and D. Salant (1999) Adoptions and Orphans in the Early Microcomputer Market *The Journal of Industrial Economics*, Vol.47(1), 87-105
10. Gandal, Neil. M (1994) Hedonic Price Indexes for Spreadsheets and an Empirical Test for Network Externalities *RAND Journal of Economics* Vol. 25, 160-170
11. Gilbert, R. J., and M. L. Katz (2001) An Economist's Guide to U.S. v. Microsoft, *Journal of Economic Perspectives*, Spring 2001 25-44
12. Gilbert, R.J. (1995) Networks, Standards and the Use of Market Dominance: Microsoft (1995), in J. Kwoka and L. White eds. *The Antitrust Revolution: The Role of Economics*, 3rd edition, Oxford:Oxford University Press.
13. Katz, M.L. and C. Shapiro (1986) Technology Adoption in the Presence of Network Externalities *The Journal of Political Economy*, Vol.94 (4), 822-841
14. Klein, B.,(2001) What can a Dominant Firm Do to Defend Its Market Position? *Journal of Economic Perspectives*, Spring 2001 45-62
15. Liebowitz, S.J and S.E. Margolis (1994) Network Externality: An Uncommon Tragedy. *Journal of Economic Perspectives*, Vol.8 n2, Spring 1994
16. Liebowitz, S.J and S.E. Margolis (1995a) Path Dependency, Lock-in and History *Journal of Law, Economics and Organization*, Vol.11, 205-225
17. Liebowitz, S.J. and S.E. Margolis (1990) The Fable of the Keys *Journal of Law and Economics*, Vol.33, April (1990), 1-25
18. Lopatka, John and William H. Page, (1995) Microsoft, Monopolization, and Network Externalities: Some Uses and Abuses of Economic Theory in Antitrust Decision-making *Antitrust Bulletin*. Vol.40, 317-70
19. Mitchell, Matthew F., and Andrzei Skrzypacz (2006) Network Externalities and Long Run Market Shares *Economic Theory* Vol.29 n3, 621-48
20. Takahashi, Takuma and Fujio Namiki (2003) Three Attempts at 'de-Winterlization' Japan's TRON Project, the US Government's Suits against Wintel, and the entry of Java and Linux *Research Policy* Vol.32, 1589-1606
21. Whinston, M.D. (2001) Exclusivity and Tying in U.S. v. Microsoft: What We Know and Don't Know *Journal of Economic Perspectives*, Spring 2001, 63-80

SUMMARY STATISTICS

Table 1: Real Prices by Year and Operating Platform

YEAR=95						**YEAR=93**				
OS	Obs	Mean	Std Dev	Min	Max	Obs	Mean	Std Dev	Min	Max
OS/2	2	323.43	1.64	322.27	324.59	2	450.73	148.24	345.91	555.56
Windows 3.X	13	333.77	168.95	84.59	595.48	19	397.01	137.42	172.68	625.44
Windows95	6	355.55	146.26	220.85	583.06	n.a	n.a	n.a	n.a	n.a
YEAR=96						**YEAR=94**				
OS	Obs	Mean	Std Dev	Min	Max	Obs	Mean	Std Dev	Min	Max
OS/2	4	182.03	110.94	63.05	316.09	1	331.33	.	331.33	331.33
Windows 3.X	8	371.20	174.27	160.57	579.66	19	373.66	157.25	87.16	612.18
Windows95	10	338.24	153.07	127.81	572.62	n.a	n.a	n.a	n.a	n.a
YEAR=97										
OS	Obs	Mean	Std Dev	Min	Max					
OS/2	4	83.64	50.27	43.53	155.47					
Windows 3.X	5	439.83	172.56	180.40	558.68					
Windows95	14	327.55	121.88	210.82	558.68					

Table 2: Hedonic Price Regression Estimates
(Dependent variable is the natural logarithm of the software's list price in real dollars LnRPRICE)
Number of observations = 107

Variable Name	**Parameter Estimate**	**Standard Error**	**Parameter Estimate**	**Standard Error**	**Parameter Estimate**	**Standard Error**	**Parameter Estimate**	**Standard Error**
INTERCEPT	**5.4354**	0.3078	**5.5796**	0.2689	5.5377	0.2023	5.55098	0.2043
Installed Base	-0.0009	0.0030	-0.0004	0.0028	-0.0007	0.0027	0.000189	0.00302
JAVA	**-0.2553**	0.0974	**-0.2486**	0.0924	-0.2543	0.0903	-0.31075	0.12729
DRAW	**0.3160**	0.1158	**0.3272**	0.1080	0.3124	0.1003	0.30623	0.10119
EDIT	**0.3696**	0.1221	**0.3586**	0.1144	0.3610	0.1127	0.34695	0.11536
DTP	0.2363	0.1857	0.2191	0.1722	0.2581	0.1447	0.24528	0.14675
PRES	-0.1152	0.2239	-0.2433	0.1917	-0.2403	0.1885	-0.23822	0.18937
BMP	0.2971	0.2895	----	----	----	---	----	----
EPS	0.0903	0.1961	-0.0115	0.1513	----	----	----	----
GIF	0.1092	0.1629	----	----	----	----	----	----
JPEG	0.1516	0.1438	----	----	----	----	----	----
TIFF	-0.1333	0.1997	----	----	----	----	----	----
Clipart	-0.0944	0.1137	-0.0465	0.1045	----	----	----	----
Multimedia	-0.2121	0.1301	-0.1877	0.1126	**-0.2016**	0.1062	**-0.22126**	0.11106
TEXT	**0.3928**	0.1126	**0.4181**	0.1066	**0.4203**	0.1035	**0.41898**	0.10394
DATA	0.2015	0.1791	**0.2986**	0.1382	**0.3004**	0.1356	**0.29576**	0.13636
RAM	0.0145	0.0106	0.0146	0.0100	0.0138	0.0097	0.01096	0.01075
Charts	-0.0902	0.1328	-0.1227	0.1209	-0.1182	0.1186	-0.11695	0.1191
OLE	0.0731	0.0923	0.1239	0.0805	0.1153	0.0765	0.11707	0.07689
DDE	-0.0936	0.0985	-0.1043	0.0950	-0.1064	0.0933	-0.10538	0.09375
YR_94	-0.0886	0.0829	-0.1105	0.0770	-0.1106	0.0744	-0.12077	0.07643
YR_95	-0.1630	0.1116	-0.1870	0.1041	-0.1902	0.1007	**-0.22132**	0.11254
YR_96	-0.1261	0.1640	-0.1706	0.1476	-0.1659	0.1450	-0.19772	0.15408
YR_97	-0.1121	0.2460	-0.1786	0.2210	-0.1655	0.2161	-0.22497	0.23657
BITMAP	----	----	**0.2595**	0.1109	**0.2656**	0.1084	**0.26998**	0.10911
WIN95	----	----	---	----	----	----	0.09021	0.14269
Condition Index	76.5500		57.6500		44.4500		45.485	
R-Square		0.9204	0.9206		0.9204		0.9208	
Adjusted R- Square		0.8639	0.8705		0.8740		0.8729	

Notes: The Regressions also included firm dummies that are not reported here in order to save space. I rejected the pooled OLS model in favor of the firm fixed effects at the one percent significance level based on results from an F-test.

NOTES